Dinesh Sapkota

Back end programming in .NET framework

Helsinki Metropolia University of Applied Sciences

Bachelor of Engineering

Media Engineering

Thesis

Date : 20 May 2013

Helsinki Metropolia University of Applied Sciences

Author(s) Title Number of Pages Date	Dinesh Sapkota Back end programming in .NET Framework 59 pages 20 May 2013
Degree	Bachelor of Engineering
Degree Programme	Media Engineering
Specialisation option	JAVA and .NET application development
Instructor(s)	Eskindir Abdela, Supervisor Kari Aaltonen, Principal Lecturer

The main goal of the project was to develop a web application which provides the services for both web and mobile client. The complete application development process was carried out by the team of four members and a supervisor. According to the interest of group members the whole project was divided into four parts; that is user interface design, mobile application development, back end development for mobile services and server side back end development of application. I got the task of server side back end development. To accomplish this task I have used SQL Server 2012, ASP.NET Framework and Entity Framework.

This project was started with designing the database structure of the application according to the project requirement. Then that structure was implemented using SQL Server Management Studio 2012. EDMX model file was generated from the existing database using ADO.NET Entity Framework to use that model as the data source for the application.

Layered Software Architecture was used to develop application by dividing the whole application into Presentation Layer, Data Access Layer, Business Logic Layer and Service Layer. C# was used as a programming language in Visual Studio 2012 to implement layered software architecture. Generic Repository Class was used to develop data access layer which reduced the use of large number of repository classes. ASP.NET controls, ASP.NET State Management and ADO.NET LINQ were quite easy to use and helpful in the process of the application development.

Keywords	SQL Server, ASP.NET Framework, Entity Framework

Contents

1 Introduction

This application was developed to participate in Microsoft Imagine Coup Competition in world citizenship category. The main purpose of this application is to solve the problems of an individual to contribute for the world citizenship without getting involved in social service organization. This application provides simple platform to help contribution for the social service.

Every day we get the news of natural disasters, people dying of hunger, children deprived of education, health, and balanced diet. But as a normal person it is difficult get access to help in this kind of situation. This application provides the platform for the individual to get connected in a social media. This application contains the information of the projects of social service organizations which need funds to complete the projects. An individual can help those projects by donating money, selling their skills and posting their task on the website. By following those projects individual can track the money they have donated and see where the money has been used. People can give both positive and negative feedback for projects. This helps other users to select the good projects. If the donation is not used according to the terms and the condition, an individual can claim that money back. It applies the same way for selling a skill: if the person selling skill did not get paid for his skill he also can claim that money back. For tasks, if the doer did not finish the task and runs away with the money then task owner also can claim the money back. For providing these entire donation, buyer and seller protection application will take 1$ for each transition.

The success of this application depends on the number of users using it. For that purpose we decided to provide services on web and mobile devices. This increases the boundary of our application resulting in easy platform for users to access the services. Providing services on mobile device, people will get an easy platform to help.

For example let us say I am in the middle of the jungle in a high way and need someone to fix my car. I can post that task by using my mobile phone. When that task is posted successfully, all the doers with that particular skill will get the notification of that task near the location of the task. After getting notification one can apply for the task by selecting the donation percentage for certain social service organization of their choice.

In this way all three parties, social service organization, task owner and dower are benefited using services of our application.

To achieve all of this functionality we have used agile software development methodology within a group of four people. The development process was started by designing system architecture based on layered application development guidelines. According to the guide lines we have divided our application into four different layers according to their distinct functionality. They are presentation layer, services layer, business layer and data access layer. In the following chapter I will discuss background and technology required to achieve the goal.

2 Web Application Development in .NET environment

Web application development should be done by analyzing the requirements, complexity to achieve the requirement and development methodology to achieve the goal. According to our application requirement we decided to follow layered application architecture and agile software development process. Layered software architecture is designed to minimize costs and maintenance requirements and it promotes usability and extendibility. Each layer should be designed with distinct functionality, specific feature and a component of layer should not know the internal details of other component. In this architecture all logical layers communicate with each other. In layered architecture the layers also communicate with clients and other applications. Since the application requirement was to develop a server side back end for web client and a mobile client, it was the perfect architecture. It increases reliability, scalability, maintainability and decreases complexity of web application. [1]

In this architecture the layer at higher level can communicate with the layer below it. The following figure shows the simple software architecture used in developing Ubuoy application. Ubuoy is the name of application which means you buoy and buy for social service. [1]

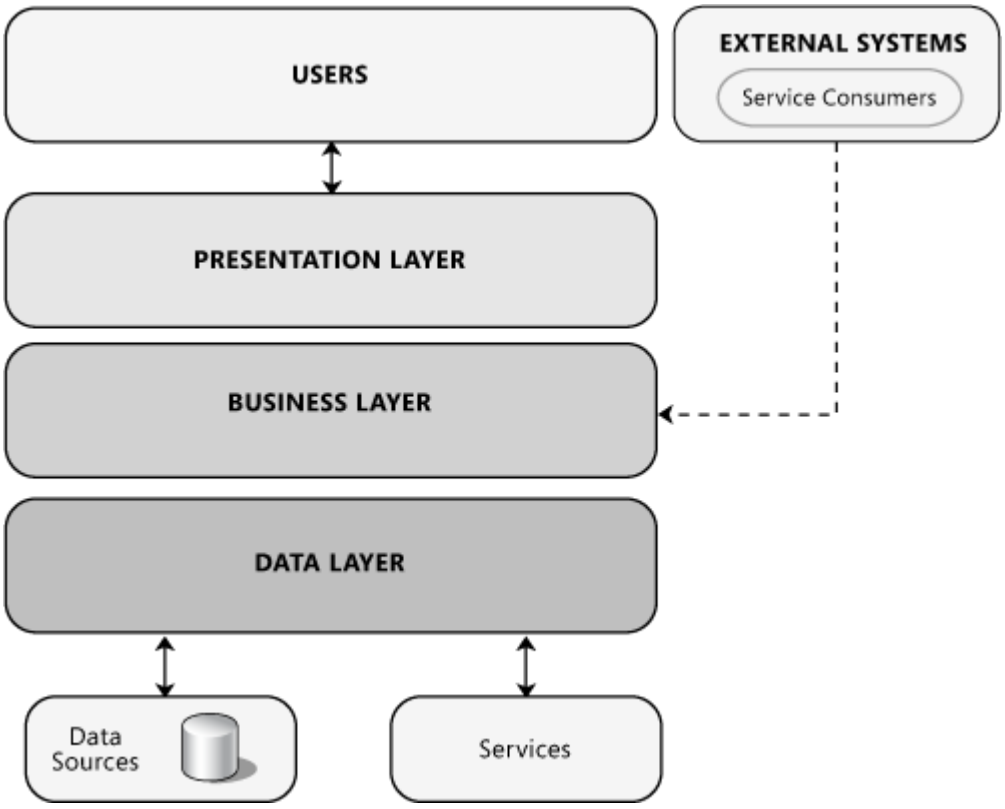

Figure 1 Common Software Architecture of web application and its components. [1]

In the above figure there are six different layers called data sources, services, data layer, business layer, presentation layer and users. Data source is a layer in which all the data are stored. To create a data source we need to create a database. Service Layer is responsible for providing services. In this project we are providing services to a mobile client. Data layer uses data sources for fetching, saving, updating and deleting data from database. Business layer communicates with data layer, and provides the methods for presentation layer for data presentation. Presentation layer contains all the logic for data presentation. User interface layer uses the data provided by the presentation layer for users. The process of developing this architecture is discussed in the following topics. [1]

2.1 Process of Database Management

World Wide Web today has evolved to be the most powerful network ever built for connecting people around the world. According to Internet World Statistics report published in 30th June 2012 there were 2,405,518,376 internet users in the world. There was a growth of 566.4% from 2000 to 2012. [2] Dynamic websites on the World Wide Web

today are able to filter the required information for the user according to the user's location, time zone, local language etc. These information entries are gathered from user computer program and website content is handled through databases. [3]

An organized collection of data divided into different tables according to their relationship is called database. This organized data can be updated, deleted, inserted and fetched by the computer program. Database Management systems (DBMSs) are applications which can interact with the user, other application and the database itself to capture and analyze data.

Complicated process of Database Management has been considerably simplified by Structured Query Language (SQL). It is capable for querying and editing information stored in a certain database management system. [3] Support for Extensible Markup Language (XML), triggers, regular expression matching, recursive queries and standardized sequences are some of important new functionalities of SQL. [3]

If the database behaviour for file storage or indexes is not well defined, then the vendors of the various SQL implementations will decide how the database will behave. Although all these implementations have the same base, but they are rarely compatible with each other. [3]

According to our web application requirement Microsoft SQL server 2012 was the perfect so, we decided to use it. When finalizing our database model, I got a task to design the database model and generate a SQL script out of it. This data model contains all the logical and physical design choices. Physical storage parameters needed to generate a design are in a Data Definition Language (DDL), which can then be used to create. Then I decided to design it on SQL Server Management Studio 2012 because it is fully compatible with Visual Studio 2012.

Microsoft SQL Server 2012

Microsoft SQL Server 2012 is like other Relational Database Management systems (RDBMSs) with server components. But the product is divided into two distinct categories business intelligence (BI) and the Database Engine.

Business intelligence

To track transformation of data in the database and use that record to make more informed business decision we can use Business intelligence. This category will be quite important for the further development of our web application. For example, fundraising Application for social service could use its data to identify project success trends, fund raising and distribution trends of organizations, region and helping patterns of its users. From that analysis, we will know our current situation, how effective is our application? What we are doing well? What should we do to be better? [4,7]

Database Engine

Database Engine is the core of the SQL Server Components. The engine starts as a service on a machine called server instance. We can run multiple instances on any server. We are connected to the server only after the server instance is created. When an application is connected, it sends Transact-SQL (T-SQL) statements to the instance. Then instance in return sends data back to the client. In connection there is a security layer that validates access to the data as specified by the database administrators (DBAs). Database engine gives us a freedom to use full capabilities of all the other components, accessing, sorting, and securing the data. Database Engines storage component is responsible for determining how data should be stored on disk. When designing our database, I specified various aspects that will dictate how our tables, indexes and views are physically organized on our disk subsystem. Storage engine is a primary component of Database Engine with additional components such as T-SQL programming interface, Security subsystem, Replication, Server Agent, disaster recovery tools, Server Integration Services and Server Management tools. [4,8]

2.2 Microsoft .NET Framework

In software development, framework is skeletal support which is pre-constructed for further development of software. It is the fundamental structure which makes the software development easier. It is a collection of software which provides the generic functionality. This generic functionality can be selectively changed by the software developer's code to provide application-specific software. It is considered as a universal reusable software platform used to develop applications, product and solution. It provides the higher level of abstraction which makes the life easier for software developer.

Framework contains support programs, compilers, code libraries, an application programming interface (API) and tool sets to enable development of a project or solution. [5, 2]

For designing and developing an application which are portable, scalable and robust (that is resist change without adapting its initial stable configuration.) will be difficult without framework supporting these features. Microsoft .NET Framework provides a lot of features. The most noticeable among them are support for cross-language runtime (CLR), just in time (JIT) compiler, support for implicit and automated garbage collection. The most important features of Microsoft .NET Framework 4 are; [5, 2]

Cross-Language Integration: Using Microsoft .NET framework 4 we can create an application that can operate using method and library written in different language. For example we can use C# to call the methods and properties of a library written in VB.NET. [5, 2]

Common Language Runtime (CLR): It is a runtime environment which supports the process like memory management, type safety, exception handling, just-in-time (JIT) compilation and automated garbage collection. [5, 3]

Portability: Portable code can be generated by compiling the source code written in a CLR language. It is intermediate machine-independent code. They are called Common Interface Language (CIL). [5, 3]

Just-in-Time (JIT) Compiler: It is a compiler which is capable of converting intermediate machine-independent code to machine-dependent code. CIL and its metadata are loaded into the memory by the CLR and JIT compiler compiles this CIL code to machine code at runtime. [5, 4]

Garbage Collection: It is the process of reclaiming the memory used by unused objects in the memory. Garbage collector is responsible for reclaiming memory occupied by managed objects when they are not needed. [5, 4]

Code Verification: Code verification enforces security by verifying the code before its execution. [5, 4]

Assemblies: It is the building block of .NET framework because they are necessary for versioning, security, deployment, and reusability of the code. An assembly consists of the metadata, the compiled CIL code and resources. Metadata contains assembly's identity information, culture information, type, dependencies and security information. [5, 4]

"ASP.NET is a language-neutral, interoperable server-side technology that allows the creation, execution and deployment of scalable web applications and services." [5, 5]

2.3 Programming Language

C# is the programming language used to develop this application. C# is pronounced as "C Sharp". C**#** is one of the .NET programming languages. C and C++ languages evolved to form C# language. It also uses the features of other programming language like Delphi and Java. When we look the very basic syntax of C# and Java they look similar. But C# code looks more similar like C++ because it inherits C++. It is object-oriented programming language which allows building reusable components for wide variety of application. [6, I]

C# requires .NET Common Language Runtime (CLR) to execute. An application which is written in C#, it executes CLR for managing memory, performing garbage collection, handling exceptions and other services without writing codes. C# compiler produces Intermediate Language (IL) and CLR converts it into machine code in memory with the help of Just-In-Time compiler and executes it. Microsoft .NET Framework Class Library contains thousands of reusable objects for C#. [6, Ii]

Since C# uses CLR it has the access to the entire FCL, there are many things we can do from it. C# can be used to create desktop applications with windows Presentation Foundation (WPF) and console application. It can be used to create ASP.NET and Silverlight applications using web services with Windows Communication Foundation (WCF). C# can also be used for assessing data both in ADO.NET and LINQ. Microsoft newest technology like Windows 8, Windows Phone 8 and Windows Azure also supports C#. [6, Iii]

2.4 Entity Framework

“ADO.NET is a set of classes that expose data access services for .NET Framework programmers. ADO.NET provides a rich set of components for creating distributed, data-sharing applications.” [7]

A collection of technologies in ADO.NET that supports the development of data-oriented software applications is called Entity Framework. The problem of developers of data-oriented applications to create model from entities and relationships is solved by Entity Framework. [8]

Entity Framework gives us an advantage of dealing data as a domain-specific objects and properties. It also gives us a higher level of abstraction when we deal with data. This higher level of abstraction helps us to create and maintain data-oriented applications with less code than in traditional applications. [8]

Entity Framework allows developers to query entities and relationships in the conceptual model (i.e. domain model) by translating those operations to data source-specific commands. This has given life to the models and freed applications from hard-coded dependencies on a particular data source. [8]

Entity Data Model Tools are used to create conceptual model from an existing database. It is used to visualize conceptual model graphically and edit conceptual model. The model is made up of a conceptual model, a storage model and the mapping between them. [9]

Modelling and mapping a model is possible in 4 different workflows,

- Code First to a New Database is the process of creating an empty database, adding new tables and defining the model using C# classes. [9]
- Code First of an Existing Database is the process of defining our model for an existing database using C# classes. [9]

In both of the workflows attributes are used for additional configuration of the classes.

- Model first is the process of creating model in Entity Framework Designer and generating database schema from that model. Model is stored in an EDMX file, which can be viewed and edited in designer. [9]

- Database first is the process of reverse engineering a model in designer from an existing database. [9]

In both of the workflows model is stored in an EDMX file, which can be viewed and edited in designer. EDMX file automatically generates the classes which we need to interact with database in our application.

Entity Data Model (EDM)

Entity Data Model is an XML file that defines a model which can be used in Entity Framework. The model is made up of a conceptual model, a storage model and the mapping between them. It also contains information required by EF Designer to render a model graphically. It is recommended to use EF Designer for creating and editing .edmx file. An .edmx file contains two types of content; [10]

Runtime Content (edmx: Runtime)

In this section there is the information used to generate model and mapping files for Entity Framework applications.

- “Conceptual Model Content (edmx:ConceptualModels): This section defines the entity types, complex types, associations, entity containers, entity sets, and association sets in the application domain. This section is written in conceptual storage definition language (CSDL).” [10]

- “Storage Model Content (edmx:StorageModels): This section describes the target database schema. and is written in storage schema definition language(SSDL)” [10]

- “Mapping Content (edmx:Mappings): This section describes the mapping between the conceptual model and the target database, and is written in mapping specification language (MSL).” [10]

.

Designer Content (edmx:Designer)

Below there is the information used by the Entity Designer to render a conceptual model graphically. It is used to define some conceptual model and design-time properties. [10]

- “Connection Content (edmx:Connection): This section describes conceptual model properties that affect the connection string. Currently, the only property you can set in this section is the MetadataArtifactProcessing property.” [10]
- “Options Content (edmx:Options) This section describes optional conceptual model properties. Currently, only the ValidateOnBuild property is set here” [10]
- “Diagrams (edmx:Diagrams): This section contains information that is used by the Entity Designer to render a graphical display of the conceptual model.” [10]

Mapping Objects to Data

It is always a challenge in Object-oriented programming to interact with data storage system. Even if the organization of classes is exactly the same with the organization of relational database tables it is not a perfect fit. Multiple normalized tables frequently correspond to a single class. The relationships between the classes are represented differently compared with the relationships between the tables. “For example, to represent the customer for a sales order, an Order class might use a property that contains a reference to an instance of a Customer class, while an Order table row in a database contains a foreign key column (or set of columns) with a value that corresponds to a primary key value in the Customer table. A Customer class might have a property named Orders that contains a collection of instances of the Order class, while the Customer table in a database has no comparable column.” [8]

“Entity Framework has tried to fill this gap called an impedance mismatch, by only mapping object-oriented classes and properties to relational tables and columns. Entity Framework maps relational tables, columns, and foreign key constrains in logical models to entities and relationships in conceptual models.” [8] Therefore, there is greater flexibility for both in defining the object and optimizing the logical model. Extensible data classes are generated from the conceptual model by Entity Data Model tools. These partial classes can be extended with additional members by the developers. The classes that are generated for a particular conceptual model is derived from base

classes. These classes provide services for serving entities as objects and for tracking and saving changes. [8]

Accessing and Changing Entity Data

Entity Framework enables application to access and change data that is represented as entities and relationships in the conceptual model. It uses information in the model and mapping files to translate object queries against entity type's representation in the conceptual model into data source-specific queries. Entity Framework manages to get the query results as an object. It provides the following ways to query a conceptual model and return objects: [8]

- LINQ (Language-Integrated Query) to Entities supports for querying entity types that are defined in a conceptual model. [8]

- "Entity SQL works directly with entities in the conceptual model and supports Entity Data Model concepts. It is used for both object queries and queries that are executed by using the EntityClient provider." [8]

"EntityClient provider manages connection, translates entity queries into data source-specific queries, and returns a data reader that the Enitiy Framework uses to materialize entity data into objects." [8] We can use EntityClient provider as a standard ADO.NET data provider to enable applications to execute Entity SQL queries and to return read-only data. [8]

The following diagram illustrates the Entity Framework architecture for accessing data.

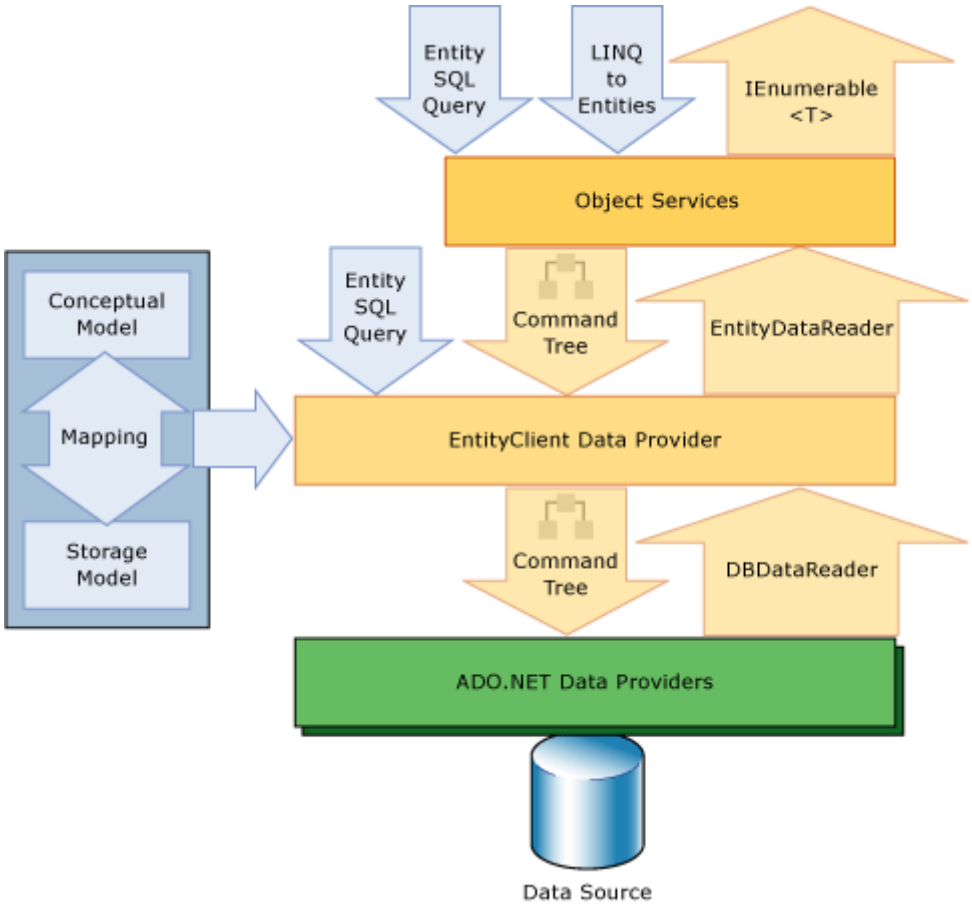

Figure 2. Entity Framework architecture for accessing data [8]

Entity Data Model Tool generates a class which is derived from **System.Data.ObjectContext** or **System.Data.Entity.DbContext** that represents the entity container in the conceptual model. This object context helps us to track changes and manage identities, concurrency and relationships. Object context class exposes **SaveChanges** method that writes inserts, updates and deletes to the data source. These changes are either made by commands automatically generated by the system or by store procedures that are specified by the developer. [8]

"The EntityClient provider extends the ADO.NET provider model by accessing data in terms of conceptual entities and relationships. EntityClient execute queries which use Entity SQL. Entity SQL provides the underlying query language that enables EntityClient to communicate with the database." [8]

2.5 Language Integrated Query (LINQ)

LINQ is a query translation pipeline which provides querying capabilities using language which runs on top of the .NET Framework. LINQ acts like a layer between the application and data store. It represents data as object and queries that data in the object model. LINQ queries are compiled at compile time and are strongly type. LINQ reduces the complexity for developing application. It allows designing and debugging query. LINQ provides unified model for accessing data in various data sources like relational database, XML files, objects and entity using syntax similar to SQL. LINQ access the data source directly. LINQ helps to get rid of impedance mismatch between programming languages and data store. [6, 226]

Operators in LINQ

LINQ provides numbers of operators for querying over collections. LINQ uses Standard Query Operators. There are two types of standard query operators. They are Standard Query Operators for IEnumarable (T) and Standard Query Operators for IQueryable (T). First type of operators operates on object that implement the IEnumarable(T) interface. It is used when working with LINQ to Objects. Second type of operator operates on the objects that implement IQueryable (T) interface. It is used when working with LINQ to SQL, LINQ to DataSets and LINQ to Entities. [6, 227]

LINQ to ADO.NET

It uses ADO.NET to connect and work with data from relational database. It is further divided into LINQ to DataSet, LINQ to SQL and LINQ to entities. [6, 233]

LINQ to DataSet

ADO.NET has an ability to explicitly cache data in a disconnected mode of operation. The DataSet is a disconnected representation of database. LINQ to DataSet helps to query data from DataSet or DataTable instances. [6, 233]

LINQ to SQL

It is used to query SQL Server database. LINQ to SQL components converts LINQ to SQL query to equivalent SQL statement which can be understood by database. DataContext should be created for translating LINQ to SQL query to corresponding vendor-specific TSQL statement. [6, 234]

LINQ to Entities

It helps us to write queries against the Entity Framework conceptual model using visual C#. LINQ to Entity converts queries to command tree queries. Then it executes the queries against Entity Framework and returns objects which can be used by both Entity Framework and LINQ. Queries for Entity Framework are represented by command tree queries that can be executed in the object context. Following process should be followed for creating and executing a LINQ to Entity query; [11, 672]

- **Constructing an object query instance:** Object query is constructed from an existing object context. Object context provides connection and metadata information that is required to compose and execute the query. [11, 673]

- **Compose a LINQ to Entities query in C#:** LINQ to Entities data source are the instances of the ObjectQuery generic class which implements generic IQueryable interface. We can specify exactly what information we need from data source. [11, 674]

- **Query Conversion:** LINQ query must be converted to a command tree representation that can be executed by Entity Framework. [11, 674]

- **Query Execution:** It is converted to the representation that is compatible with Entity Framework then it is executed against data source. [11, 674]

- **"Materialization:** It is a process of returning query results back to the client as CLR types. LINQ to Entities query result data records are never returned. There is always a backing CLR type defined by the entity framework or generated by the compiler." [8]

2.6 ASP.NET State Management

Each request from the client browser to the web server is understood as an independent request in ASP.NET because it is based on stateless HTTP protocol. State management is used to maintain state information across the multiple requests. ASP.NET Framework has the build in support for both server and client side. [6, 24]

Server-Side State Management

The technique used to store state information on the server side is called server-side state management. Application, Session and Cache objects are used to store state information on server-side state management. [6, 24]

Application Object

It stores the data which is accessible to all the users in an application. Objects stored in the application state are accessible from all the modules of application. They are available as long as the application is running. [6, 24]

Session Object

Session is defined as the duration of connectivity between the client and server. Session objects are used to store user specific data until the session is active. Web server creates and mentions the session. Web browser sends cookie that contains session identifier and all request when session is started. IIS web server uses this session ID to identify request belonging to a particular session. If IIS web server could not find any session ID then it will generate session ID along with the request. Session timeout can be specified in web.config file. Session ID is stored until the browser instance is unchanged, even if the session object expires after a specified timeout. If an application does not store anything in the session state, a new session state is created with each request. In web.config it is possible to choose a session with or without cookies. If the session uses cookies then session ID is stored inside cookies. In a cookieless session, session ID is embedded in the URL itself. If cookieless session is used then application

will be supported in all browsers whether cookies are enabled or disabled in a browser. [6, 25]

Session State Storage Modes

Session state can be stored in InProc or State Server or SQL Server storage modes.

Storing Session State in the InProc Mode

It is the fastest mode of session state storage. It stores session data in ASP.NET worker process. But the performance is also affected on the amount of the data stored. The session state stored in the inproc mode depends on the application domain. If the application domain restarts then the session state will be lost. [6, 26]

Storing Session State in a State Server

“State Server mode uses a stand-alone Microsoft Windows service which is independent of IIS and can run on a separate server.” [5, 27] This storage mode decreases the performance due to the process of serialization and deserialization of objects. This storage mode has its own process and memory. It is stored in external process due to which crash of ASP.NET will not affect the data stored in it. This helps to share information across web garden or web farm. The application which contains multiple worker process is called web garden. Web farm is the process of using multiple servers to host the application and dividing the traffic among them. [6, 27]

Storing Session State Using SQL Server

Storing session state which uses SQL server provides reliable, secure and centralized data storage. This session storage stores session data in database table of SQL Server after serialization. It is used in web farms. It reduces the performance of serialization and deserialization of the data stored and retrieved from SQL server. SQL Server Mode is more secured because Server Security can be configured. [6, 27]

In distributed web server environment storing object types will degrade performance due to serialization and deserialization. So it can be a good idea to store basic types in

session state. We should not use Response.Redirect method after setting session in login page because it calls Response.End method which stops the execution of page and session ID is lost. FormsAuthentication.RedirectFromLoginPage method can be used to save our session. [6, 30]

Cache Object

Caching can be used to improve the performance of the application. We can store frequently used data in the main memory. This data can be used to serve incoming requests, reducing network traffic and reducing use of server resource which results improved performance. Saving data in the cache memory is much faster than retrieving it from database. Caching can be done in three different ways in ASP.NET. [6, 31]

Page Output Caching

It stores the entire page in the memory which makes easier to respond for the same page request by fetching the data from cache. When there is new request for the page, runtime checks if the requested page exists in the cache. If it exists then it loads the page from cache otherwise the page is rendered dynamically. It is useful for the pages which are static and does not change for some interval of time. [6, 31]

Partial Page Caching

It allows us to cache the certain portion of the page. This is useful when we have a page of both static and dynamic contents. [6, 32]

Data Caching

It helps us to store data in the cache which can be retrieve later which reduces the load on the server. [6, 32]

Cache Expirations

Cache expiration policy is used to refresh the cache by keeping it sync with the data store. Cache expirations are time-based, file-based and key-based. [6, 33]

Time-Based Expiration: It defines a specific time period to store the page in cache. When that time is spent the item is removed from cache. [6, 33]

Client-Side State Management

We can use ViewState, hidden fields, query string and cookies for client-side state management.

ViewState

It is used to store the state of an ASP.NET pages because they move back and forth. It does not store the controls in the page. It stores control ID and their corresponding values which can be lost due to the postback to the server. ViewState represents the state of the page when it was last processed on the web server. View state is the property of all server controls stored with key-value pair using System.Web.UI.StateBag. ViewState is enabled for all server controls but we can enable or disable it at page, control, application and machine lavels. [6, 35]

It is a good choice to store small amount of data in ViewState but if it contains large amount of data it decreases the performance of the application. We can remove runat = “server” tag completely from the form if our page does not uses postback to the server. ViewState can be secured by encrypting its contents. [6, 37]

Cookies

Cookie is a text file stored on the client side which is used by the browser to store textual messages. Data are stored as name-value pair separated by equals sign. It is stored in the Cookies directory on the system. A temporary cookie exists in the memory as long as the user session is alive. Permanent cookie is stored in a physical location on the client system and it is deleted after it expires according to the client browsers setting. Cookies can be created, read and deleted using request and response object. Since cookies are saved in the client browser, they are not safe to store sensitive data, as they can be viewed, edited and deleted by the client. [6, 43]

3 Technologies

For building the application we should use technology according to the compatibility and simplicity to use. I decided to use SQL Server Management Studio 2012, Visual Studio 2012, Team Foundation Server and Agile Scrum Methodology. The reason behind selecting these technologies is described in the following topics.

3.1 SQL Server Management Studio 2012

SQL Server Management Studio 2012 is an integrated environment for database management. It also contains tools for configuring, monitoring and administrating instances of server. It consists of large number of graphical tools and script editors which helps developers to access server. Both graphical tools and script editors can be used to create database. It supports all the components of SQL Server. [12]

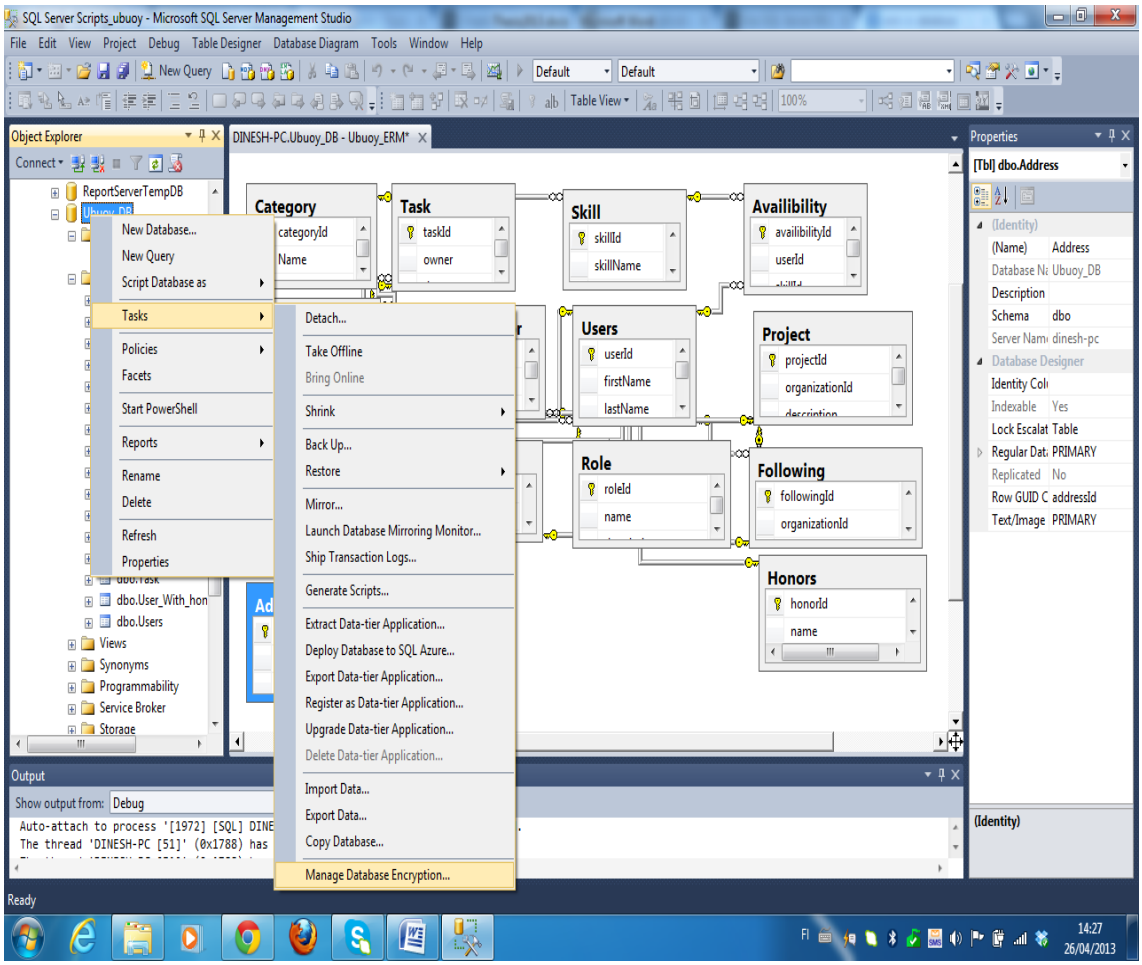

Figure 3 User Interface and tools of SQL Management Studio 2012

Using management studio I have created database called Ubuoy_DB with 13 different tables with different relationship. Management studio can be used to create entity relationship model from database or vice versa. It helps us to detach, backup, restore, database encryption, ship transaction logs, generate script and deploy database. Due to all these features I used this technology for my application. [12]

3.2 Visual Studio 2012

Visual studio provides the tools for designing, developing, debugging and deploying applications. It can be used to develop desktop, web and mobile application for Microsoft. According to the application requirement we need an Integrated Development Environment (IDE) which supports ASP.NET Framework, Entity Framework, Windows Communication foundation, C# language and Team Foundation Service (Version Control System). It has the features like code editor, debugger and designer. It contains tools like properties editor, object browser, solution explorer, team explorer, data explorer and server explorer. Due to these reasons I decided to use it.

3.3 Team Foundation Service

Version control system is used to provide a base version of application for team of developer. Team foundation service is a Microsoft version control system which can be used from visual studio. In a visual studio there is a tool called team explorer, which helps us to connect to the team foundation server (TFS). We can view and map the applications in TFS from source control explorer. After mapping the project in local folder, file can be check out for editing. If user checks out one file then it cannot be edited by other user which prevents overwriting of code. It can be edited by other user only after checking in that file to TFS. We can work offline with the local file and then we can check in the changes. All of the checks in are recorded with user name and date and time. There is a merge tool which helps us to merge the changes while checking in. While getting the latest version from TFS there can be some conflicts and it is handled by the resolve conflict tool. If the conflict is not resolved automatically then we need to resolve it manually. It gives us the option to take server version or keep local version which helps user to choose the version he needs. Since, this application was developed by a group of four people, TFS helped us quite a lot by saving our source code on a cloud which can be retrieved whenever it is needed.

4 Software Architecture

“Software architecture is the process of defining a structured solution which meets the entire technical and operational requirements. It is used in optimizing common quality attributes like performance, security and manageability. To design the system architecture we need to have series of decisions based on a wide range of factors. Each of these decisions can have considerable impact on the quality, performance, maintainability, and overall success of the application.” [1]

Software architecture is described as the structure of a system. Where, system is the collection of the components that accomplish a specific function. In other words, Software Architecture is a process of organizing all of its components to support specific functionality. The following figure illustrates common application architecture with components grouped by different areas of concern. [1]

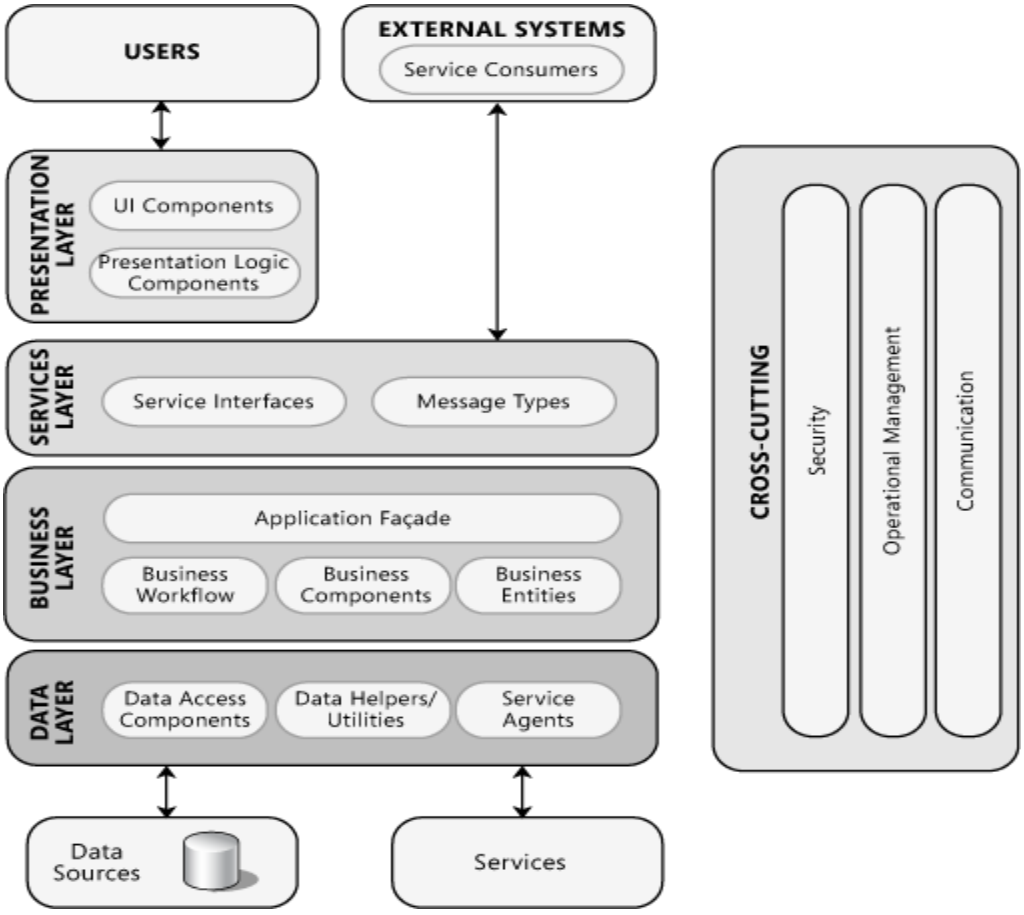

Figure 4 Common Software Architecture of web application and its components. [1]

After grouping of components, we need to focus on the interaction between the components to make it work together.

The key principles for designing system architecture which minimizes costs and maintenance requirements, promotes usability and extendibility are as follows

- **Separation of concerns.** Application should be divided into distinct features to prevent the unnecessary overlap in functionality. [1]
- "**Single responsibility principle.** Each component should be responsible for only a specific feature." [1]
- "**Principle of Least Knowledge.** A component should not know about internal details of other component." [1]
- **Don't repeat yourself (DRY).** Specific functionality should be implemented in only one component. It should not repeat in another component. [1]
- **Minimize upfront design.** Design should be based on what is needed. Especially for agile development, avoid big upfront design. Because in agile development our design evolve over time. [1]

4.1 Data Access Layer

Data access layer provides simplified access to data stored in a database. DAL returns a complete reference to an object with its attributes instead of a row from the database table. This helps us to create client modules with higher level of abstraction. By using this layer we can retrieve and write database easily. Insert, delete and update commands could be executed with a simple function. This layer hides database from the external world. For each table in the database we need to create an interface and repository classes to create data layer. In software engineering these two files are used to access data from the database.

Repository Pattern

Domain-driven design (DDD) is a process of developing software for complex application by connecting the implementation of software to an evolving model. This concept was discovered by Eric Evans in his book "Domain Driven Design" published on 22,08,2003. One of the major patterns in DDD is the repository pattern. Repository pattern connects the application database and business solution. [13]

The word repository came from Latin word repositorium, which means a vessel or chamber in which things can be placed or a place where things can be collected. Information technology it is a central place in which data is kept and maintained in an organized way using computer storage. Repository is a place from which specific database, files or documents are obtained for relocating or distributing in a network. Repository is the aggregation of data itself into some accessible place of storage or some ability to selectively extract data. [15]

Repository is a collection of resources that can be accessed to retrieve information from database. Repository hides the detail of how exactly the requested data is being fetched/persisted from/to the database. It creates the query satisfying the supplied criteria and returns the result set. It allows all of our code to use objects without having to know how the objects are persisted. [15]

The repository pattern is an abstraction layer. It provides a well-organized approach to maintain a separation between an applications data access layer and business layer.
It gives an advantage of making code more maintainable, readable and testable. Basically it adds a separation layer between the data and domain layers of an application.

The domain-specific objects and properties provided by the Entity Framework Model can be used to create repository classes. We need to create repository classes for each domain specific object. Entity Framework Model creates domain specific objects and property from the tables and relationships in the database.

To create a Data Access layer in C# we need to have a repository interface which contains the definitions of related functionalities that a repository class can implement. Repository class basically implements create, read, update and delete (CRUD) functionalities. Following figure shows the repository interaction between the client and the data source.

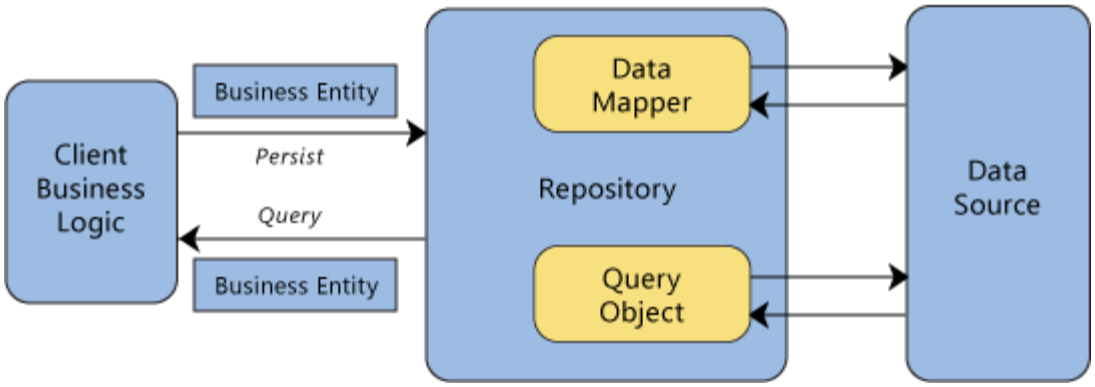

Figure 5 Interactions of Repository. [15]

If the client submits request to save information in the database table from the business logic through repository class to the data source and send the response to the business logic.

Generic Repository

Repository class is directly proportional to the database model Entity. For a complex application there can be large numbers of entities. To create a business solution for that application we need to implement large number of repository classes with same CRUD functionalities. This results in coding the same stuff in large no of class decreasing productivity.

Entity Framework 4.0 contains the IObjectSet interface which makes data objects modification possible and the ObjectContext class which has a generic CreateObjectSet<TEntity>() method. Due to these two things, creating generic repository class is possible in entity framework. To create DAL, we need generic repository interface and its implementation class. After having all the functionality implemented in a generic repository we can use it, to access all the entity in the database model by passing the entity name to this class. Basically, it acts as a specific repository class for that entity, which is passed to the generic repository class from Business Logic Layer. [14]

4.2 Business Logic Layer (BLL)

BLL is used in software system architecture to separate direct dependency of presentation layer to DAL to increase the scalability and maintainability of the software. Business Logic Layer represents business objects corresponding to the entity objects in database. This layer provides methods for presentation layer to access the data on the database. In this layer we can implement security for the server side of our application, by validating inputs received from the presentation layer and prevent SQL injection from the client inputs. If our application business requirement changes in time then it is easy to implement this change by just changing our business layer. In this approach we can leave both data and presentation layers same which makes our application easily maintainable. During the release of application we can just release BLL into the production environment. [16]

There are some things which we need to consider before designing the business layer. This is a job of software architect to design BLL, by separating tasks into all the different areas of concerns to minimize complexity in implementing it. For example, logic for processing business rules, business workflows, and business entities all represents different areas of concern. The components of one design must focus on its area of concern and should not contain the codes of other areas of concerns. There is a Microsoft guideline for designing BLL and we should consider them to design software architecture which reduces the complexity of implementing it. Those guidelines are as follows; [16]

- **Decide if a separate business layer is needed**. It is always a good idea to use a separate business layer where there is a possibility to improve the maintainability of application which we are designing. [16]

- "**Identify the responsibilities and consumers of business layer**. This will always help to decide what tasks a business layer must accomplish, and how to expose our business layer to presentation layer. Business layer can be used for processing complex business rules, transforming data, applying policies and for validation. If business layer is used by presentation layer and external application then business layer should be exposed through a service." [16]

- **Do not mix different types components in business layer.** Prevent mixing presentation and data access code in business logic code. Separate

business layer logic from both presentation and data access logic and testing of business functionality. Use centralized common business logic functions to promote reuse. [16]

- **Avoid tight coupling between layers.** Use certain level of abstraction to minimize coupling when creating an interface for the business layer. We can take an example of message based interface between the presentation layer and the business layer. [16]

Security and Reliability of application in BLL

There are seven different issues which we need to consider before designing to prevent the mistakes. They are the following issues,

Authentication

The process of identifying an individual based on the username and the password is authentication. Authentication strategy must be implemented in business layer for security and reliability of our business application. If authentication is not implemented properly in an application, then application will be vulnerable to spoofing attacks, dictionary attacks, session hijacking and other types of attacks. To prevent this while designing following authentication strategy must be followed. [16]

- Avoid authentication in business layer if it will be used only by a presentation layer and service layer on the same series of layer in a trusted boundary. [16]
- Use separate user stores and implement a single Sign-on mechanism if business layer will be used in multiple applications. In this case we should always use built-in platform mechanisms whenever possible. [16]

Authorization

The process of giving access right to an individual to access system objects based on their identity is authorization. This access rights are given according to the role of the

user defined in a database. It is important to define effective authorization strategy for the business layer to make application secure and reliable. If an application fails to do so then it is vulnerable to information discloser, data tempering and elevation of privileges. Following guide lines should be followed to design an authorization strategy. [16]

- Protection of resources can be done by applying authorization to callers based on their identity, account group, roles, or other contextual information. For roles, consider minimizing the level of considered in roles as far as possible to reduce the number of permission combinations required. [16]
- Avoiding delegation whenever it is possible will increase performance and scaling opportunities of application. [16]
- Prevent mixing authorization code and business processing code in the same component. [16]
- Since authorization is typically spread throughout the application, prevent authorization infrastructure to impose any significant performance overhead. [16]

Caching

The process of storing items, data objects, parts of a page, in memory at the initial time when they are requested to increase performance and scalability of web application is called caching. This information can be stored on the web server, software request stream as proxy server or browser. Avoid recreating information that satisfies a previous request which reduces the processor load and increase the performance of web application. In ASP.NET there is two types of caching is possible, they are output caching and application data caching. Output caching allows storing dynamic page and user control responses on any HTTP cache-capable device in the output stream, from the originating server to the requesting browser. On similar request the page or user control is not executed and cached output is used to satisfy the request. [16]

Appropriate caching strategy should be used in business layer for the performance and responsiveness of the application. Use caching to optimize reference data lookups, avoid network round trips and avoid duplicated processing. Caching strategy should be decided to load the cache data. Following guidelines are followed for designing a caching strategy, [16]

- Static data that will be reused regularly should be cached within the business layer. Consider caching data that cannot be retrieved from the database quickly and efficiently but avoid caching vary large volumes of data that can slow down processing. [16]

- Caching of data should be done in a ready to use format within the business layer.

- Sensitive data should not be cached and if sensitive data is cashed then create the mechanism to protect sensitive data in the cache.

Coupling and Cohesion

Coupling is a measurement of dependency of one business layer to another. It is always good to reduce the coupling between the business layers. [16]

Cohesion is the process of measuring how close the members of the module are related to other members. [16]

Before designing the components for business layers, we should ensure that they are highly cohesive and loosely coupled between each other. This increases the scalability of the application. Consider the following guideline when designing for coupling and cohesion. [16]

- Circular dependencies should be avoided. Business layer should know only about the data layer but should not know anything of presentation layer and the external applications accessing the business layer. [16]

- Abstraction must be used to implement loosely coupled interface. [16]

- Use tight coupling within the business layer unless dynamic behaviour requires loose coupling. [16]

- Promote high cohesion. Always avoid mixing data access logic with business logic in your business components. [16]

- Message- based interface should be used to expose business components to reduce coupling. [16].

Exception Management

Consider using exception handling in the situations where the system can recover from an error. This provides a means for control to be returned from a function to the program. [16]

Designing an effective exception management solution for business layer makes the application secured and reliable. If an application is unable to handle exceptions, it is vulnerable to Denial of Service (DoS) attack leading to reveal sensitive and critical information about application. For designing an exception management strategy following guideline should be considered. [16]

- Only internal exceptions should be catch because they can be handled. For example, catch data conversion exceptions which are occurred when trying to convert null values. Exceptions should not be used to control the business logic or application flow. [16]

- Design an appropriate exception propagation strategy. For example, allow exceptions to catch up to the boundary layers where they can be logged and transformed as necessary before passing them to the next layer. [16]

- Ensure to catch an exception in an appropriate place so that it will not catch on other place. Clean up the resources and state after an exception occurs. [16]

- “Design an appropriate logging and notification strategy for critical errors and exceptions that logs sufficient detail from the exceptions and does not reveal sensitive information.” [16]

Logging, Auditing, and Instrumentation

Logging, auditing and instrumentation are required in business layer to improve the security and reliability of application. Failing to do so can leave your application vulnerable to repudiation threats, where users deny their actions. Log files may also be required to prove wrongdoing in legal proceedings. Auditing is generally considered most authoritative if the log information is generated at the precise time of resources access, and by the same routine that access the resource. Instrumentation can be implemented using performance counters and events. System monitoring tools can use this instrumentation and other access points to provide administrators with information about the state, performance, and health of an application. The following guidelines must be considered when designing a logging and instrumentation strategy; [16]

- Logging, auditing and instrumentation must be centralized in a business layer. Third party solutions such as the Apache Logging Services can be used for exception handling and logging features. [16]

- Should include instrumentation for system critical and business critical events in business components. [16]

- Business sensitive information should not be stored in the log files. [16]

- Logging failure should not affect normal business layer functionality. [16]

- Consider auditing and logging in access to functions within business layer. [16]

Validation

The process of checking the data provided by the function of presentation layer according to the properties of data provided is called validation. Effective validation for business layer is important for the usability and reliability of the application. Failure in effective validation results the application open to data inconsistencies and business rule violations and poor user experience. It also leaves application vulnerable to security issues such as cross-site attacks, SQL injection attacks, buffer overflows and other

types of input attack. Following guidelines should be considered when designing a validation strategy; [16]

- Enable validation of all input and method parameters within the business layer, even if the input validation is implemented in the presentation layer. [16]

- Validation approach should be centralized to maximize testability and reuse. [16]

- Consider that all user input is malicious and validate all user input data for length, range, format and type. [16]

4.3 Presentation Layer

In layered application design presentation layer contains the components that implement and display the user interface and manage user interaction. This layer includes controls for user input and display and additional components that organize user interaction. Presentation layer contains interface components and presentation logic component. User interface components are the application's visual elements used to display information to the user and accept user input. Presentation logic is the application code that defines the logical behavior and structure of application.

There are several common issues that should be considered before designing it. These are those common areas; [17]

Caching

Caching is used to improve application performance and UI responsiveness. Caching in the presentation layer optimizes data lookups avoiding network round trips. It is used to store repetitive processes to avoid unnecessary duplicated processing. Following guidelines can be considered when designing caching strategy; [17]

- Keep cached data in a format which is ready to use when working with in-memory cache. [17]

- Do not cache sensitive data without encrypting it.[17]

- Business logic should not be created depending on the data cache because they might not be in cache any more. For example in business transaction I might want to fetch most recent data to apply to the transaction rather than using old data stored in the cache. [17]

- Implement authorization rights for cached data. Cache data in such a way that the data are accessed by the user if he has got the authorized role to access the data. [17]

- All the access to the cache must be thread safe. [17]

Navigation

Navigation strategy should be designed in a way that the user can navigate easily through your screens by separating navigation from presentation and UI processing. Navigation links and controls should be used in a consistent way to reduce user confusion and to hide application complexity. [17]

Validation

An effective input and data validation strategy is critical for the security of an application. User input validation rules must be determined in the presentation layer. Following guide lines can be used for data validation strategy; [17]

- Input validation should be handled by the presentation layer and business rule validation is done by business layer. If business and presentation layers are physically separate, business rule validation logic should be mirrored in the presentation layer to improve usability and responsiveness. This can be achieved by using common validation rule components in both layers. [17]

- Validation errors must be handled correctly by avoiding expose of sensitive information in error message. Validation failures should be logged to assist in the detection of malicious activity. [17]

4.4 Services layer

Service-based solution are composed of multiple services, each communicating with the others by passing messages. In application services are the components seen and used by the users. When an application provides services to other applications it must implement features which support clients. The common approach to develop application which provides services to another application is to use a services layer.
Service layer is designed to use highest level of abstraction, which is possible after grouping functionality into layers. Public interface must be defined for each layer depending on the application using it. After defining the layers and interfaces the application should be deployed to use its services. The interaction between the layers and tiers of other applications is possible after choosing a communication protocols. In agile development in the beginning the service is simple and it evolves with time. Following design steps must be considered before designing the web services. [18]

- Layering strategy must be chosen before designing the service layer. Layers must be separated on the basis of distinct roles and functionality. Layering improves maintainability of the application with easy scalability to improve performance. [18]

- Distribution of layers and components must be done wisely to prevent mistakes. In an application if the presentation layer components used busyness layer components synchronously then service layer should deploy the business layer and presentation components on the same physical tier to maximize performance. [18]

- Interaction between the layers should be done in a predefined rule to reduce circular references. If there are two layers with the dependency on the other layer then there is circular dependency and it reduces the performance and scalability of application. [18]

- It is good to use collapse layer for the application with very limited business rules like pulling the data from the web service and displaying that data. It may make sense to have a service layer to have validation rules to serve the data. [18]

- Cross cutting concerns should be identified after defining layers. After identifying the cross cutting concerns we can design separate components to manage these concerns. This will help to achieve better reusability and maintainability. [18]

- It will help to enforce loose coupling between layers by defining interface for a layer. This helps to hide the internal details of layers. [18]

- Communication protocol should be used to improve the performance, security and reliability of the application. It is more important to design our application in distributed deployments. [18]

5 Application Development Process

Our project requirement was to provide the services to buy and sell skills by donating certain percentage of the money to the social service organization. To make these services reliable and secure needed a lot of discussion. After discussion we decided to track the complete process of buying, selling and donation. Our project requirement was also to provide the feedback services from users and social service organization which will help the users to select an honest buyer and seller of skill and social service organization.

Since this is a charity based application with financial transactions it forced us to think about security and transparency of the application. For transparency of the application we decided to make the whole process of transaction track able by the user who is directly involved in the process. We decided to give honors to the users according to the positive feedback received and amount of donation provided to the social service organization.

The project requirement was to achieve this entire requirement, dynamically through the web service. This is the process I followed to create the software architecture for the web application. First of all I decided to follow the layered software architecture development process to achieve my goal. In layered software application development process we can divide the whole application into different layers according to the distinct task they are performing.

To achieve the project requirement I decided to divide my web application into data source, services, data access layer, business layer, presentation layer, users and external system. Since this application was supposed to provide services to the mobile client, we can consider it as an external system.

5.1 Data Source Development Process

My task was to create the server side back end for the application. According to the application requirement I should be able to store large amount of information in the organized collection. All the process of saving, updating, fetching and deleting of data in an organized collection should be done by the application.

Database

The organized collection of data being stored in logically divided tables, according to the relationship among them, is called database. Database table is a set of data values which is organized using a model of vertical columns and horizontal rows. The point of intersection between the rows and column is called cell. A table has specified numbers of columns but it can have any number of rows. Each row is identified by the unique key index called ROWID. ROWID is an address of row which is always unique and set as primary key.

Database Management System

The system designed to save, update, fetch and delete data in database table according to its relation is called database management system. In our application development process we designed our database management system discussing with all the team members in a meeting. It is always good to design the database system after the group discussion because it makes our view broader which helps to analyze data in a broader prospective.

In our database design we divided the tables according to the distinct nature of data and its relationship with the data stored in another table. We decided data types to be used for storing data according to nature of data stored in the column. Primary key is a

unique identifier called ROWID, which is used to identify the set of data stored in a row. Primary key column is used to identify each set of data stored in database table. Foreign key column in a table stores primary key value of the other table according to its relationship with that table. We decided to use Globally Unique Identifier (GUID) data types for primary and foreign key for the security of data in the database table. This helps us to make our database safe from information disclosure attack. [20, 457]

When the database design was ready I got a task to implement that design in SQL Server 2012. For that purpose I used an IDE called SQL Server Management Studio 2012 because it supports all the functionality of SQL Server and it was fully compatible with Visual Studio 2012. It provides large variety of tools to design database. Using this IDE we can easily create database and its tables with relationships by using both graphical user interface and using SQL query language. Using graphical user interface we can define the connection strategies, I have used windows authentication strategy for connecting to my database. Security, user role, server logs and triggers can be enforced for database by using graphical tools while creating the database. Management studio helps us to deploy our database in the cloud services. Database in the management studio also can be used locally by connecting to the visual studio. Database can be created by executing SQL query in management studio. To create a simple database we can execute a simple query like this;

```
CREATE DATABASE Ubuoy_DB
```

Figure 6 SQL statement for creating Database

After that I grouped all the data which were supposed to be stored in the database according to their category in separate tables. All the tables were connected with each other according to their relationships by the help of primary and foreign key in the tables. We can create tables in the database by executing the SQL query shown in the following lines.

```
CREATE TABLE [dbo].[Task] (
    [taskId]      UNIQUEIDENTIFIER CONSTRAINT [DF_Task_taskId] DEFAULT
(newid()) ROWGUIDCOL NOT NULL,
    [owner]       NVARCHAR (50)    NULL,
    [doer]        NVARCHAR (50)    NULL,
    [startedOn]   DATETIME         NULL,
    [endline]     NVARCHAR (MAX)   NULL,
    [deadline]    DATETIME         NULL,
    [description] NVARCHAR (MAX)   NULL,
    [categoryId]  UNIQUEIDENTIFIER NULL,
    [status]      NVARCHAR (MAX)   NULL,
    [skillId]     UNIQUEIDENTIFIER NULL,
    [updateDate]  DATETIME         DEFAULT (getdate()) NULL,
    [money]       VARCHAR (MAX)    NULL,
    CONSTRAINT [PK_Task] PRIMARY KEY CLUSTERED ([taskId] ASC),
    CONSTRAINT [FK_Task_Category] FOREIGN KEY ([categoryId]) REFERENCES
[dbo].[Category] ([categoryId]),
    CONSTRAINT [FK_Task_Skill] FOREIGN KEY ([skillId]) REFERENCES
[dbo].[Skill] ([skillId])
);
```

Figure 7 Task Table create statement

This code is used to create Task table which has a relationship with the category and skill tables. This table contains 12 columns with different data types according to the data to be stored in this table. This process was repeated to create all the required tables. After implementing all the tables and relationships, required database structure was achieved. It is shown in the following figure.

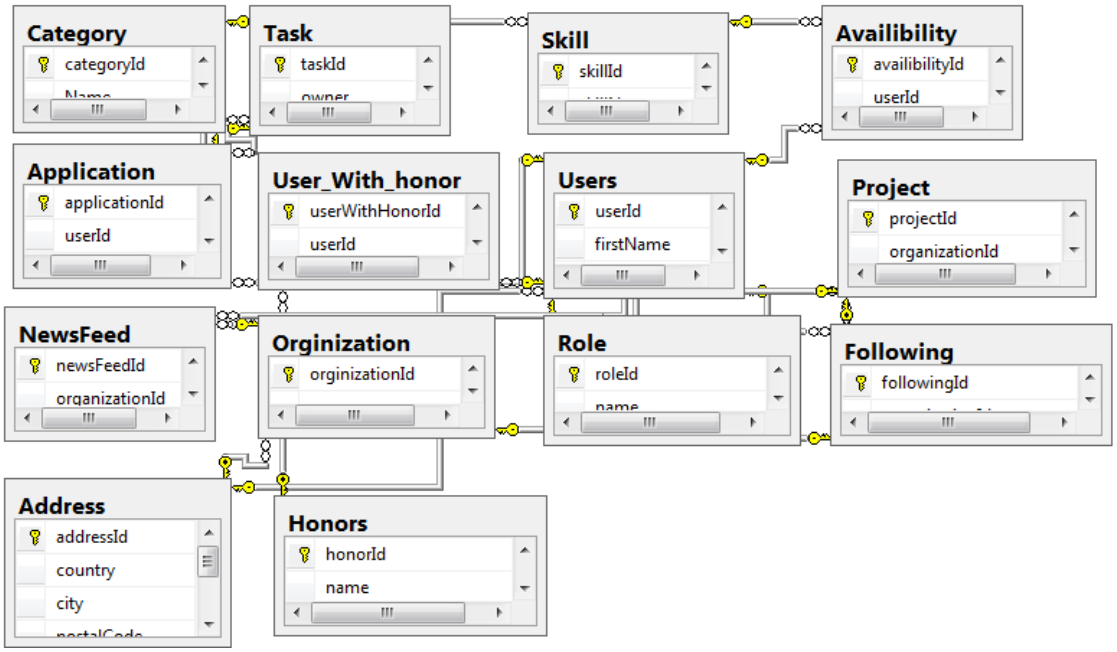

Figure 6 Database structure of Ubuoy Application

According to our project requirement this was the initial database structure I designed.

After implementing database, the next step is to connect that database from a server to use it. For that purpose I used ADO.NET Entity Framework because it helps us to separate our application from the relational or logical model by providing a layer of abstraction on top of relational model. When the database was ready, I created the EDM from the existing Ubuoy database by using Entity Framework. This model is automatically created by the framework which can be edited and updated using designer. This EDM consists of a collection of entities, entity types, entity sets and their relationship. EDM is used as a data source for this application. During this process a connection string is created by the framework and it is stored in the web.config file. This connection string is used by the application for performing CRUD operation to the database. The following piece of code is a connection string generated by the framework;

Figure 7 Connection String generated by entity framework

```
connectionString="metadata=res://*/Model.UbuoyDbModel.csdl|res://*/Model.Ubuoy
DbModel.ssdl|res://*/Model.UbuoyDbModel.msl;provider=System.Data.SqlClient;pro
viderconnectionstring=&quot;datasource=(LocalDB)\v1.0;attachdbfilename=|DataDi
recto-
ry|\Ubuoy_DB_Model.mdf;integratedsecurty=True;MultipleActiveResultSets=True;Ap
```

This connection string is used by the application to connect to the database and to interact to the database table.

5.2 Data Access Layer Development Process

When the data source of Ubuoy application was ready I designed a data access layer for my application. This layer is used to access the data stored in the database. In application development process we use repository classes to achieve this goal. It is the best practice to create an interface repository class which contains method signature and constant declarations. That interface is implemented in all the repository classes. In normal application development process we create separate repository class for each database table. When developing an application with large numbers of tables, programmers end up writing large amount of similar code which decreases the productivity of the development team.

There is a new concept of creating a generic repository class which can be used as a specific repository class by passing the entity name to this class at run time. To implement generic repository class first I created an interface for that class. In the following lines there is a code for creating interface class;

```
public interface IRepository<T>: IDisposable where T :
class
    {
        IQueryable<T> Fetch();
        IEnumerable<T> GetAll();
        IEnumerable<T> Find(Func<T, bool> predicate);
        T Single(Func<T, bool> predicate);
        T First(Func<T, bool> predicate);
        void Add(T entity);
        void Delete(T entity);
        void Attach(T entity);
        void SaveChanges();
        void SaveChanges(SaveOptions options);
    }
```

Figure 8 Code used to create interface of generic repository class

In the above code interface class uses generic variable T which can be any entity and it is defined as IDisposable to release the allocated resources. All the required method signatures and constants are declared in this class.

After that I created a generic repository class to implement this interface. The interface class is implemented as follows;

Figure 9 Implementation of interface in generic repository class

```
public class GenericRepository<T> : IRepository<T> where T : class{}
```

In place of T we pass the entity name while calling the method of generic repository class and during the run time the compiler generates the repository class of that specific entity to get the desired output.

Due to the support of IObjectSet interface and ObjectContext class with generic CreateObjectSet<TEntity>() method in Entity Framework 4.0, it is possible to implement generic repository class. I have implemented all these features of entity framework as follows;

```
private ObjectContext _context;
private IObjectSet<T> _objectSet;

public GenericRepository(): this(new Model.Ubuoy_DB_ModelEntities()){}

public GenericRepository(ObjectContext context){
            _context = context;
           _objectSet = _context.CreateObjectSet<T>();
```

Figure 10 Implementation of Object Context and Object Set

In the above code first of all I have initialized object context and object set then the entity container name Ubuoy_DB_ModelEntities is used as a datasource for repository classe. Then the objectContext and the objectSet are used to implement generic method CreateObjectSet<T>(). After this it is ready to implement CRUD operations. In CRUD operation I have used exception handling for getting rid of exception in runtime and defend the application DoS attack.

First of all I have implemented Find() method in my generic class which returns a collection of data which satisfies the given condition. It is one of the most important methods in the CRUD operation. Following lines of code shows how it was implemented:

```
public IEnumerable<T> Find(Func<T, bool> predicate){
            try{
            return _objectSet.Where<T>(predicate);
            }catch (InvalidOperationException ex)
           { return null; }
       }
```

Figure 1 Find method implementation in repository

In the above piece of code I have made this method public, which makes this method accessible from business layer class. It returns the collection of data so that I have used IEnumerable<T> which satisfies the given condition. In programming the given condition is called predicate. I have used try catch statement for exception handling which prevent the crash of the application if any exception occurs in the run time returning the null value. It also helps the application from DoS attack.

In Add() method the values of entity is obtained from the business layer and it was pushed into the database table using this method. Following code shows how it is implemented:

```
public void Add(T entity){
            if (entity == null){
            throw new ArgumentNullException("entity");
            }
             _objectSet.AddObject(entity)
            }
```

Figure 12 Add methods in repository class

This method is used to save entity and we do not need to return any values so that void is used. By using frameworks objectSet, AddObject() method we can save the entity into the database table. For updating the data in the table we can use framework Attach() method. All the other implementation is similar to Add() method. Delete() method is also similar, the only difference is that one has objectSet, DeleteObject() method. It is used to delete record in the database.

To persist the add, update and delete operations in the database we need to implement and call the SaveChanges() method every time after performing these operations. An example of save changes is shown in the following code:

```
public void SaveChanges(){
             _context.SaveChanges();
            }
```

Figure 13 SaveChanges method

In the above method objectContext, SaveChanges() method is used to persist the changes in table. This method ensures that the changes have been saved.

When dealing with the database from the server side, it is always important to release resources after the desired operation is achieved because it increases the performance of the application. In the following code it shows how to use Dispose() should be used;

```
public void Dispose(){
            Dispose(true);
            GC.SuppressFinalize(this);
        }
```

Figure 14 Dispose method

When we call this method from the business layer it triggers the private method in the repository class by passing the bool value of the method and it releases the resources. In the following code the process is shown;

```
protected virtual void Dispose(bool disposing){
            if (disposing){
            if (_context != null){
            _context.Dispose();
             _context = null;
              }
            }
        }
```

Figure 15 Dispose method implementation

Above code uses the ObjectContext Dispose() method to release resources of the context and sets it value to null.

After implementing all of these my data access layer is ready for server to implement CRUD operations. This complete layer is implemented using Entity Framework which has separated my database and data access layer by providing the higher level of abstraction and speed up whole application process. Due to the implementation of one generic repository class it has reduced large no of code in my application. This helped me to concentrate on the higher layer of the application development. When the data access layer was ready, I started working on the business logic layer.

5.3 Business Logic Layer Development Process

BLL is used to provide methods for the presentation layer to access the data on the database. This is the most important part of the application development process because in this layer all the business logic of the application is implemented. This layer is used to implement security for the server side of application by authentication of user, validating the user inputs and prevent SQL injection from the client inputs. Security is enforced in this layer to preserve the application from middle man attack because if they bypass the security in the presentation layer, this layer prevents the attack. This layer separates the DAL and presentation layer, which makes the application mention-

able and reusable. If our business logic changes, we can make changes in this layer and the other layers remain the same.

I have created different classes for each entity in the database which helps in separating the business logic to improve the maintainability of the application. BLL is used for processing complex business rules, transforming data and applying policies for validation. After having different classes for each entity it is easier to avoid tight coupling between the layers. BLL classes are created in such a way that they do not have any detailed knowledge of the DAL classes. Business layer can just use the methods in the DAL without knowing the entire class and this is the basic principle of application development in layered architecture. [19]

In this section I am going to discuss how to implement the classes in the business layer. After designing and finalizing the classes to be used, I started implementing BLL to access data from DAL. First of all I started initializing the generic repository in a business layer by providing the entity value. It enables the generic repository class to act as a repository class of that specific entity. Initialization can be done as follows:

```
private GenericRepository<Category> categoryRepository;
```

Figure 16 Initialization of category repository

In the above code we pass entity Category in generic repository class to get category repository class in run time. After this we need to construct a constructor to achieve that functionality. In the code below it shows how to construct the category repository;

```
public CategoryBusinessObjects(){
            this.categoryRepository = new GenericRepository<Category>();
            }
```

Figure 17 Constructor for generic repository

After this constructor has been created we can use it to implement the CRUD operation by using this repository to access the methods in DAL. I am going to discuss how to insert a record in the database table using this class. The following piece of code shows how to implement that:

```
public bool AddCategory(string name, string description, string image){
            categoryRepository.Add(new Model.Category() { Name=name, De-
scription=description, image = image });
            categoryRepository.SaveChanges();
            return true;
        }
```

Figure 18 AddCategory method in BLL

This method is used by the class of presentation layer to pass the values of name, description and image. When this method is called from the presentation layer with all the required parameters it calls the Add() method in the repository class by passing entity category with its values. In repository class this values of the entity is saved in the object context. After calling the method of object context SaveChanges(), these values are saved in the database table.

Validation

Since the server side validation should be implemented in this layer I have created a validation interface in this layer called IValidation.cs. This class implements two validation components as follows:

```
List<string> ValidationSummary { get; set; }
bool IsHavingValidInputs(params object[] inputs);
```

Figure 19 Validation interface components

These two parameters are used by the business layer to implement the validation. In this layer we validate all the inputs from the user to enforce the security in the application. In the following code we can see the step by step implementation of input validation;

```
public List<string> ValidationSummary { get; set; }
```

Figure 20 Validation summary variable

This code is used to initialize the validation summary as a list. Then we should implement the constructor for validation summary. The following code shows how it is done:

```
public UserBusinessObjects(){
            userRepository = new GenericRepository<User>();
            this.ValidationSummary = new List<String>();
            }
```

Figure 21 Initialization of validation summary

After this we can use this summary for our input validation. For validation of the inputs certain validation rule should be enforced. According to those rules I am going to discuss some of those methods which were used in the validation. The following piece of codes how to check required fields validation;

```
private bool RequiredElementsInPlace(object[] inputs){
            foreach(var input in inputs){
             if (string.IsNullOrEmpty(input.ToString()))
                    return false;
             }
            return true;
            }
```

Figure 22 Required element validation method

In the above method all the inputs provided by the presentation layer are checked and if any of the value is null, returns false which triggers the error message and the CRUD operation is not executed. The following piece of code checks if the email input is in the correct format;

```
private bool EmailIsInCorrectFormat(string p){
            if (p.Contains("@")){
            var halfEmail = p.Split('@')[1];
              return halfEmail.Contains(".");
            }
            return false;
        }
```

Figure 23 Email format validation method

The above method checks the validity of email input. All the validation methods are called from IsHavingValidInputs method. Then it decides whether to send those to DAL or not.

The following code shows how it is implemented:

```
public bool IsHavingValidInputs(params object[] inputs){
       bool isValid = true;
       var requiredElements = new object[] { inputs[0], inputs[1],
inputs[2], inputs[3], inputs[4] };
       if (!PasswordsMatch(inputs[1].ToString(), in-
puts[2].ToString())){
            ValidationSummary.Add("Passwords do not match");
                isValid = false;
            }
       if (!EmailIsInCorrectFormat(inputs[0].ToString())){
            ValidationSummary.Add("Email not in correct format");
                isValid = false;
            }
       if (!RequiredElementsInPlace(requiredElements)){
             ValidationSummary.Add("All fields are required");
                isValid = false;
            }
            return isValid;

            }
```

Figure 24 Input validation method

If the inputs from the presentation layer satisfy all these conditions then only these val-

```
public bool RegisterUser(string email, string password, string password2,
string firstName, string lastName, string gander, DateTime dob,string im-
age){
            var validation = string.Empty;
      if (this.IsHavingValidInputs(email, password,password2, firstName,
lastName, gander, dob)){
      if (!UserEmailExists(email)){
            userRepository.Add(new Model.User() { email = email, password
= password, firstName = firstName, lastName = lastName, gender = gander,
DOB=dob, image=image, userId = Guid.NewGuid() });
             userRepository.SaveChanges();
             return true;
                }
                else return false;
            }
            else
            {
                ValidationSummary.Add(validation);
                return false;
            }
        }
```

ues are sent to the DAL. The following piece of code shows how to do that;

Figure 25 Validation in user registration method

This user registration will be successful if the method IsHavingValidInputs() returns true, otherwise it will ask user to make correction through the error message.

Authentication

I have used user authentication process to provide access to the application on the basis of username and password provided by the user. It is used in an application for the security and reliability of the application. If the application does not use the authentication in an application, then the application is venerable to spoofing attacks, dictionary attack and session hijacking attack. The process of user authentication is implemented as follows:

```
public bool UserExists(string userName,  string password){
            var exists = userRepository.Find(x => x.email.Equals(userName)
&& x.password.Equals(password));
            return exists.Count() > 0;

            }
```

Figure 26 User authentication method

In the above code this methods gets the username and password value from the presentation layer and it checks if the values exists in the database table. If they exist, it returns the count which is used by the presentation layer to provide access to the application.

LINQ to Entities

For accessing data from DAL LINQ to Entity was used in the application development. Standard query operators are used to query IEnumerable<T> interface or IQueriable interface in LINQ. In the following piece of the code we can see how it is implemented;

```
public IEnumerable<Task> GetLatestSixTask(string parent){
             return taskRepository.Find(x
=>!string.IsNullOrEmpty(x.owner)).OrderByDescending(x =>
x.updateDate).Take(6);
           }
```

Figure 27. LINQ query operators' example.

The above method uses IEnumerable<Task> as LINQ query with standard query operators like OrderByDescending() and Take(). LINQ query helped me a lot to deal with database in my application.

5.5 Presentation Layer Development Process

The final step on development of the Ubuoy application was to build the presentation layer. It is responsible for the interaction between the user and the application. This layer is designed to respond to the user's action in the UI. It contains UI components and the logic used to present information to the user. It defines the logical behavior and structure of the application. In layered system architecture it can access the method in BLL to provide interaction between the user and application.

The project requirement was to provide the services to the users according to their location, skills, interest and setting made by the user. Presentation logic was designed to meet all these criteria. To generate the content for user dynamically was the only solution for this problem. Due to this, all the classes in the presentation were implemented to generate the content of page dynamically.

Profile page implementation is one of the examples of the dynamic content generation in this application. All the contents of the profile page are user specific which is generated according to the information of the user stored in the database table. At the time of user login, when the login is successful a user session is created and all the required information of the user is stored in it. This session is used to generate the content. The following code shows how to create the session;

```
Session["LoggedIn"] = true;
Session["UserId"] = user.userId;
Session["UserRole"] = user.Roles.SourceRoleName;
```

Figure 28 Assigning the values for session variables

In the above code a session is created and different values for the session variable are assigned. When the user successfully logs in, the UserId and UserRole is saved to provide the services to the users according to his/her id and role. All of this information is saved in the session until the session is valid.

I have used this UserId value of session to retrieve user object from the database for presenting the user specific information in user interface. Due to this requirement I ended up in creating the whole UI dynamically. In this application development process I was busy in designing the back end of the application and my colleague was busy in developing the UI. At this point we were supposed to implement these two different types of programming to work together. There was one big problem of understanding these two different codes by the other person who has not written it. We had also problems because I did not have good understanding in UI designing and Cascading Style Sheets (CSS) coding. Then we decided to work together for developing this part of application. Skype provided us an environment to work together by shearing each other's screen. It was a great experience of coding together by helping each other. It helped us to continue our application development process by learning new things in each other's code.

We started the implementation of UI and presentation layer from our profile page of the application. The presentation logic of profile page was to display user projects and module according to his involvement and the recent activity in application. The main problem in this part of implementation was to generate exactly the same HyperText Markup Language (HTML) designed by UI designer from code behind part of aspx class. The code below shows the process of HTML code from code behind:

```
HyperLink mainContent;
mainContent.CssClass = "tile quadro double-vertical image border-color-
LightGrey";
mainContent.Attributes.Add("data-role", "tile-slider");
mainContent.Attributes.Add("data-param-period", "3000");
mainContent.Attributes.Add("data-param-direction", "left");
mainContent.ID = "ProjectTrue"+count+"";
mainContent.Attributes.Add("oncontextmenu", "return false;
event.preventDefault();");
```

Figure 29. Dynamic UI creation code example

In the above code I have defined CSS class for mainContent with its all attributes and the ID value to generate the HTML code which uses all these information during the runtime. When we run this code it generates the following code;

```
<a id="PageRegionContent_ProjectTrue4" class="tile quadro double-
vertical image border-color-LightGrey"
onclick="event.preventDefault();" data-role="tile-slider" data-
param-period="3000" data-param-direction="left"
oncontextmenu="return false; event.preventDefault();"
```

Figure 30. HTML code generated by the application.

In the above code we can see that a hyperlink was created with the ID, class and attribute's information provided in the code behind. All the browsers use this HTML code to show the contents of the web page. Complete UI of Ubuoy application was developed from the code behind files.

All the data for the presentation logic was obtained by calling the method of business logic layer. The process of accessing method is shown in the following code;

```
private Project _userProjects;
var ProjObj = new BusinessLayer.ProjectBussinessObjects();
 _userProjects = ProjObj.GetProjectById(usersProjectId);
```

Figure 31. Process of accessing the method of business layer.

In the above code project object and projectBusinessObjects class are initialized to call the method GetProjectById(). The value returned by the method is stored in _userProjects object. This object is used to provide data for the presentation layer.

I have used user controls to display different contents in a same web page according to the client's selection. To achieve this feature, I have passed the control ID in query string whenever the client requests for a page. Using this control ID, proper page is selected and displayed for the user. This process is shown in the following code:

```
var userControlName = Request.QueryString["formId"].ToString();
            switch (userControlName)
            {
                case "AddProject":
                    addProject.Visible = true;
                    Page.Title = "Add a Project";
                    Label formHaderLabel = (La-
bel)Master.FindControl("formHaderLabel");
                    formHaderLabel.Text = "| Add a Project";

                    break;

                case "AddSkill":
                    addSkill.Visible = true;
                    Page.Title = "Add a Skill";
                    Label formHaderLabel2 = (La-
bel)Master.FindControl("formHaderLabel");
                    formHaderLabel2.Text = "| Add a Skill";


                    break;
```

Figure 32. Selection of user control according to the value passed in query string.

In the above code value passed in the query string is obtained and checked by using switch. For example if AddProject is the value pass in the query string, then it satisfies the first case. This sets addProject user control to be visible, which displays the content of addProject in the web page.

Site navigation is also one of the important parts of presentation layer. All the pages of our web application are not accessible by the user without login. There are certain pages which contain sensitive information of the user. To secure these pages from the man in middle attack I have checked the session variable. If the session exists, the sensitive page like profile page is displayed and if the session does not exist, the user is redirected to the login page. The process of implementing this feature is shown below.

```
protected void Page_Load(object sender, EventArgs e){
            if (Session["LoggedIn"] != null){
             //do stuff to load the page
             }else{
                 Response.Redirect("~/Login.aspx",false);
            }
             }
```

Figure 33. Checking Session variable in page load method.

ASP.NET application lifecycle provides a page load method, which is called for loading the content of the page. In this method I have checked the value of session variable; if it is null, it redirects to the login page. If the session variable is not null, it means the session is created. Then user detail is checked and the profile page of that user is displayed.

Client side validation was done by using ASP.NET input validation controls like compare validation control, required field validation control, range validation control and regular expression validation control. For example I have used RegularExpressionValidation Control for checking the email format. In the property window of this validation control we can choose validation expression manually, which is shown in the following figure 34;

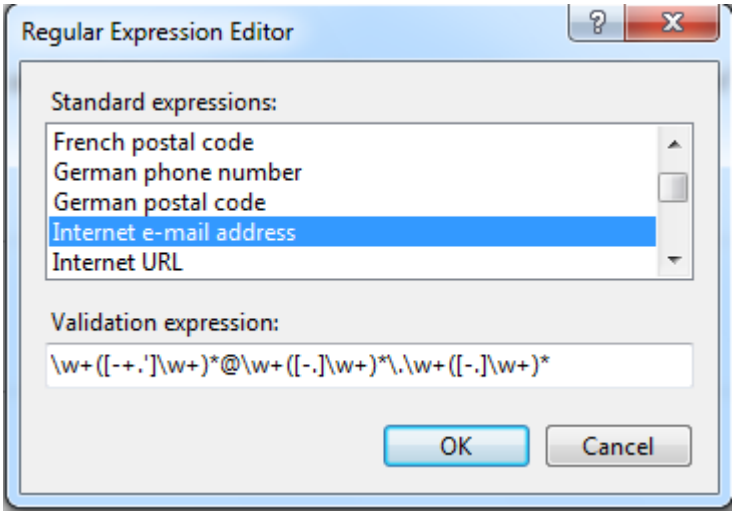

Figure 34. Regular Expression dialog box.

From the above dialog box for email validation I chose the option Internet e-mail address for checking the format of the email address. After doing this I will get the ready-made method implementation for email validation. This validation control is hooked up with the email text box to check user input before the postback event is called. Client side validation reduces the server load and helps the user to fill in correct data in the input field.

After implementing all these required features in the classes of presentation layer, our application was ready for testing and user review.

7 Verification and Result

It is always difficult to develop an application in the estimated time period because it is dependent on factors such as nature of the application under development, team size, requirements and personal experience of the team members. After careful consideration of these factors, our team selected SQL Server 2012, Visual Studio 2012, Entity Framework, ASP.NET Framework, Team Foundation Server and layered software architecture as tools for the application development. At the end of the application development process I realised that we made the right selection of the tools.

Our team was able to accomplish the estimated functionality of the application before the deadline. Due to this, I was quite happy about the team performance in the application development. It was possible due to the continuous help from the supervisor and hard work of the team members.

I have created the server side back end of the application including the presentation logic of the application. Since it was an application with complex requirement, there are some bugs in this application. I did not have enough time to implement the functionality which helps to increase the performance of the application and client side validation of the inputs provided by the user. It was a great learning experience because I have never built a complex application like this during the school courses. Implementation of the generic repository class was difficult in the beginning but in overall it also speeded up the development process. Generating the UI content dynamically was also a new experience for me and at the end of this process I learned its importance in web application. Team foundation server provided a platform to use the same version of the application to all team members. I had problems while checking in the changes after removing files from the local version stating that some files were missing from the local version. Many times the database file in the server version was not updated even after checking in the local version. Null reference exception and object reference not initialized were the most common problems I had in the process of development.

The performance of the application was the same in Google chrome, Mozilla firefox and Internet explorer browsers. Layout of the web application in Internet explorer and Mozilla firefox were same and it was slightly different in Google chrome. Due to the layered software architecture of the application, it is easy to develop the application further. The application can be further developed by continuing from its current state.

Due to the use of Entity framework, modification of the database also became easier. Windows 8 metro user interface was successfully implemented by the presentation layer, which is quite a new concept in web application development.

Our team successfully provided the platform for the social service organization and the individuals for helping each other by buying and selling a certain skill. However, the application is not yet complete because the actual transaction of the money has not been implemented and it is the most important part of the application.

8 Discussion

This project was successful due to the right selection of tools for the development of the application. It was developed by following agile methodology, and by having the scrum meeting on a weekly basis. In each meeting our supervisor used to give the task for the coming week. In each meeting we used to discuss about the weekly task, problems faced while doing the task and the best solution for that problem. We used to get the task for the next week. This application development methodology helped us to get results every week and our application was evolving week by week. The meetings also helped us to learn from the others work because we were having the discussion about; How the task was done? What were the problems? What were the solutions to that problem?

When I created a database design in SQL Server Management Studio, I had problems in using that SQL script generated from the database. Then I discussed about that problem with my supervisor and there was a problem due to the script created by the management studio which used the windows authentication validation method of my computer. Due to that, I was unable to execute the script on a school computer. By using the authentication strategy of that specific computer, I was able to create the database in the school computer.

After generating the model from the existing database by using Entity framework in Visual Studio 2012, I had problems in using generic repository class. It was a quite new class for me and I had never created this kind of class before. Since it was s new concept, there were not sufficient materials on the web. Then I received help from the mentor of our project to implement this class. I would not have been able to implement generic repository class in this project without his help. It was also useful because it

reduced a considerably large amount of code to be written in the DAL. During the whole application development process whenever there was a problem he did his magic to get us going.

While designing the business layer, I suffered with the problem where I did not see any error in the code but when I ran the application it crashes with two exceptions called null reference exception and object reference not initialized. Then I searched for the problem and I found the solution. Null reference exception appeared because I was using the object without initializing it or I was using the null object. Object reference not initialized was caused while using the object without initializing it to an instance.

In designing the presentation layer I was supposed to use dynamic UI creation but I was completely unaware what is going on in the user interface designed by another developer. Then we had a great problem to continue the development because we ended up in the situation that we were unable to continue the development without working together. After having a discussion with the user interface designer, we decided to work together but time was another problem because we were free only during the evening. Then we decided to use Skype screen shearing technique as a solution where we can see what the other person is doing. After working for two days together in the Skype, we started to understand the code written by the other person. It was fun working together and it also speeded up our process. In presentation layer the code written by me should interact with the code written by the UI designer. Because of this, both of us worked together to create the complete presentation layer. If we had not done this there would not have been any possibility of getting the result which we have now. This was the most complex part of our application. Then we realised that if all of the group members have done the same during the whole process we might have developed a complete application for deployment. By doing all of this we met the objective set in the beginning of the application development process.

Because we followed agile software development process, our application is flexible and compatible for the evolution of application without making any changes in the current state of application. All of this was quite easy due to the use of entity framework and layered software architecture.

Due to the lack of research regarding the technique which should be used to provide buyer’s and seller’s protection, we have not implemented the actual transfer of money

in this application. It is the most important part of the application which should be secured and a single mistake in this part influences the future of the whole application. We are doing the research at the moment to implement this part of the application. This application is also not completely functional in the mobile device. There is also some work left in the development of mobile client.

In my opinion this application development process might be useful for developing any kind of application because it supports further development and each layer can be changed according to the business requirement without affecting the other layers. Microsoft Developer Network (MSDN) helped me quite a lot during the whole process because of its quality of the documentation and code example.

9 Conclusions

The main goal of the project was to create a web application by providing a platform for social service organization and individuals to help each other by buying and selling a certain skill. These functionalities have been successfully achieved using the technique explained in this document. The tools I have used helped me a lot to speed up the development process. MSDN library was the main reference of the whole development process because it is the Microsoft certified developer's library with the certified documentation, code syntax and code example. Without ASP.NET framework and Entity framework, it would not have been possible to develop this application in 5 months. Layered system architecture and the generic repository class are the main strength of this application.

Due to the use of Entity framework, database modification and update was easy by using generate and update model from database features of the framework. An automatically generated database model file which was generated by the framework was easy to use just as the data source. A large variety of the classes offered by the ASP.NET framework library made the creation of the application easier. Layered system architecture provided reusability, scalability and reliability of the application. Agile application development methodology followed in this project helped me quite a lot to get this result by evolving the application on the weekly basis. For the further development of this application, some research should be done in providing buyers' and sell-

ers’ protection, securing the transaction of money and enhancing the usability of the application throughout the process.

References

1 MSDN: Chapter 2: Key Principles of Software Architecture [Online]
http://msdn.microsoft.com/en-us/library/ee658124.aspx
Accessed on 17/04/2013

2 Internet World Statistics website[Online]
http://www.internetworldstats.com/stats.htm
Accessed on 28/03/2013

3 What is SQL?[Online]
http://www.ntchosting.com/databases/structured-query-language.html
Accessed on 28/03/2013

4 Patrick LeBlance. Microsoft SQL Server 2012 Step by Step on 2013. Microsoft Corporation: California; 2013

5 Cristian Nagel, Bill Evjen, Jay Glynn, Karli Watson and Morgan Skinner. Professional C# 2012 and .NET Framework 4.5. November 6 2012

6 Joydip Kanjilal. ASP.NET 4.0 Programming. The McGraw-Hill Company, 2010

7 MSDN: ADO.NET [Online]
http://msdn.microsoft.com/en-us/library/e80y5yhx.aspx
Accessed on 29/03/2013

8 MSDN: Entity Framework Overview [Online]
http://msdn.microsoft.com/en-us/library/bb399567.aspx
Accessed on 29/03/2013

9 MSDN: ADO.NET Entity Data Model Tools [Online]
http://msdn.microsoft.com/enus/library/vstudio/bb399249(v=vs.100).aspx
Accessedon30/03/2013

10 MSDN: Entity Framework EDMX [Online]
http://msdn.microsoft.com/en-us/data/jj650889
Accessed on 02/04/2013

11 Tony Northrup and Mike Snell .Web Applications Development with Microsoft .NET Framework. Microsoft Press: Redmond, Washington 2010

12 MSDN:SQL Server Management Studio [Online]
http://msdn.microsoft.com/en-us/library/hh213248.aspx
Accessed on 26/04/2013

13 Domain Driven Design and Development in Practice[Online]
http://www.infoq.com/articles/ddd-in-practice
Accessed on 09/04/2013

14 Creating generic Entity Framework 4.0 repository [online]
http://geekswithblogs.net/seanfao/archive/2009/12/03/136680.aspx
Accessed on 27/04/2013

15 MSDN: The Repository Pattern [Online]
http://msdn.microsoft.com/en-us/library/ff649690.aspx
Accessed on 09/04/2013

16 MSDN: Chapter 7: Business Layer Guidelines [Online]
http://msdn.microsoft.com/en-us/library/ee658103.aspx
Accessed on 18/04/2013

17 MSDN: Chapter 6: Presentation Layer Guidelines [Online]
http://msdn.microsoft.com/en-us/library/ee658081.aspx
Accessed on 18/04/2013

18 MSDN: Chapter 6: Service Layer Guidelines [Online]
http://msdn.microsoft.com/en-us/library/ee658090.aspx
Accessed on 19/04/2013

19 Creating generic Entity Framework 4.0 repository [online]
http://geekswithblogs.net/seanfao/archive/2009/12/03/136680.aspx
Accessed on 27/04/2013

20 Jesses Liberty. Building .NET application with C#. O'Reilly Media: February 2005